

# Going NUTS for Network Testing

Tim Raphael

Regional Product Line Manager  
IP Network Automation

The Nokia logo is centered within a large, stylized graphic on the right side of the slide. This graphic consists of two concentric circles. The outer circle is white, and the inner circle is a dark teal color. The word "NOKIA" is written in white, uppercase letters across the middle of the inner teal circle. The background of the entire slide is a green-to-teal gradient, with the circular graphic partially overlapping it.

NOKIA

# The Questions

1. What is testing?
2. What are the different types of testing?
3. How does testing apply to network design and operations?
4. How can we test the network?

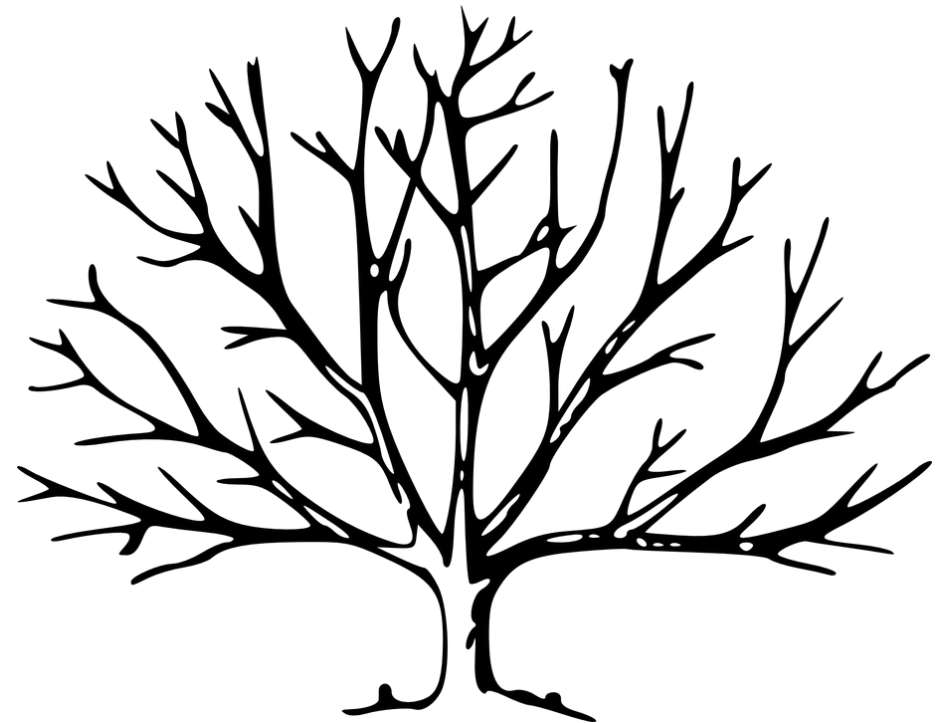
# What is Testing?

# What is Testing?

*Software testing is the process of evaluating and verifying that a software product or application does what it's supposed to do.*

(<https://www.ibm.com/topics/software-testing>)

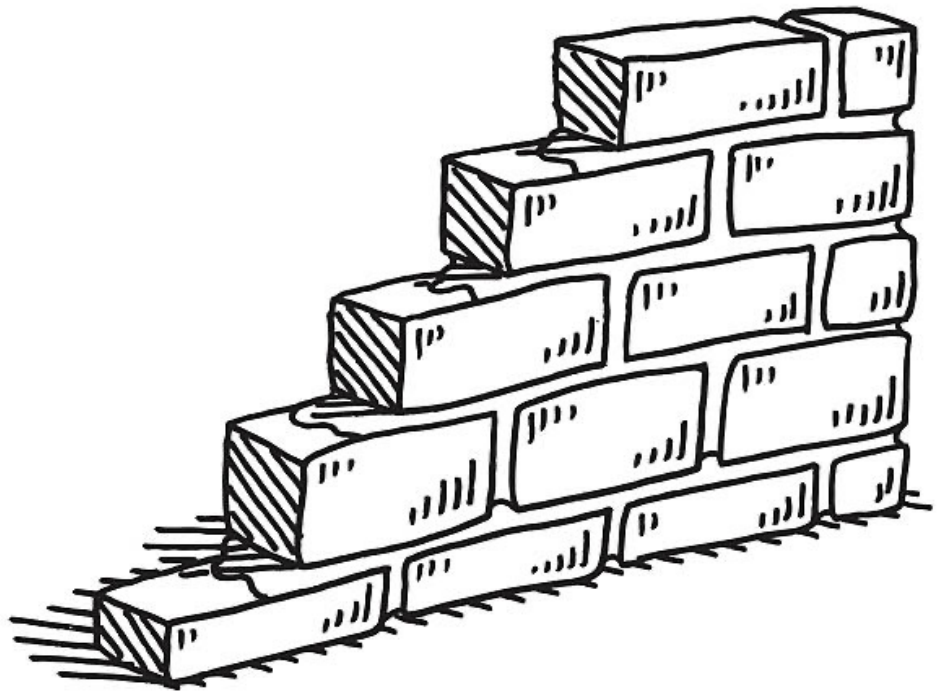
- Taking a system, giving it known inputs and inspecting the output.
- Systems, especially software can be infinitely complex with lots of branching logic.
- Testing coverage is about ensuring that all branches of a system are traversed in the testing process, ensuring all “code paths” are tested.



# Types of Testing

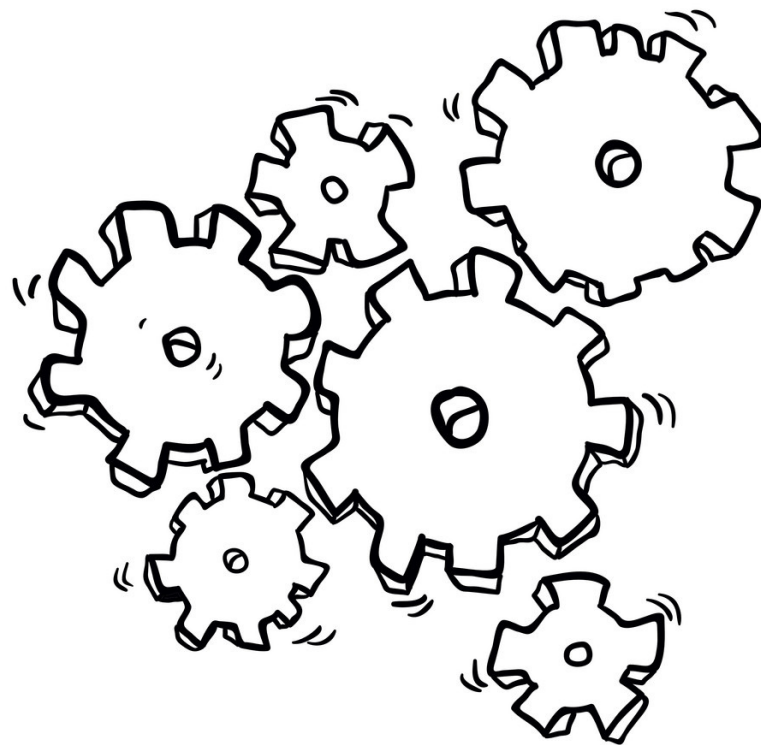
# Unit Testing

- Designed to test the smallest components of a system in isolation.
- Removes dependencies on other parts of the system and carefully controls the inputs.
- E.g. A method to take some JSON and return a BGP Peer configuration stanza. Known input = Known output.
- Stubbing and Mocking are common in unit tests to ensure isolation.



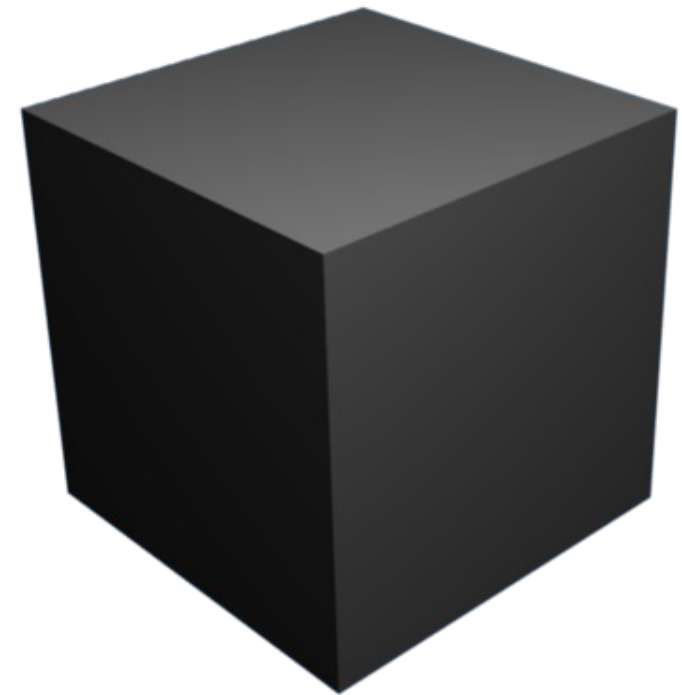
# System Testing

- Tests on a complete system, usually used to test end-to-end function of a given feature-set.
- Also referred to as “E2E” or “Integration testing.”
- Examples:
  - Given a device name and a known set of inputs, a system can generate a complete device configuration (that’s correct).
  - Given a device name, a system can act against a device and disable all BGP neighbours.



# Blackbox Testing

- Blackbox meaning we're not able to introspect the internal implementation details.
- In the functional sense, this is about externally interacting with the system passing a known input and inspecting the output.
- E.g.
  - Testing a vendor's NOS is typically Blackbox Testing. Little control over the internals, only a promise of input → output.





# How does testing apply to Network Design and Operations?

# Typical Network Design Use-Cases

- Migrating between vendors
- Migrating from one technology to another
  - RSVP + MPLS → SRv6
  - VPLS over MPLS → EVPN-VXLAN
- Design to a pre-defined specification
  - Internal Specification, RFC, IETF, IEEE, TMF, BBF...
- Iterative Design aka “Figure it out as we go”

# Software Development Thinking

- All these processes could benefit from software-development thinking.
- Design Validation
  - Proof-of-Concept validation against a specification → Unit Testing
  - An end-goal is met after a given iteration → Unit Testing
- Implementation Validation
  - One network does the same thing as the other → Systems Testing
  - One technology does the same as the other → Blackbox Testing

# What about Operations?

# Network Baseline

- Networks tend to have a baseline of expected functionality (known outputs)
  - Certain interfaces up with protocols enabled
  - Certain learned protocol states or neighbours
  - Certain traffic levels
- As Network Engineers we often “feel” this – why not codify it?
- Why not capture network state and check it against a different reality! → [Unit Testing](#)

# Software Upgrades

- Software Upgrades start with...
  - Known, good, state then...
  - Structured changes are implemented...
  - We expect a similar (possibly different) end state!

→ System Testing

# Customer Impact

- Making changes to underlying infrastructure that affects customer services.
  - Transport mechanism, technology or device might change
  - Customer service is expected to stay consistent

→ Blackbox Testing

# How to test the network?

# NUTS – Network Unit Tests

- Open Source Software that enables unit testing using Python's excellent pytest framework.
- Enables network-focused use-cases by stitching Nornir with a programmatic test framework.
- Comes with pre-packaged NAPALM-integrated testing examples.
- Well documented examples with easy customisation.
- Ability to drop to native Python for advanced cases.

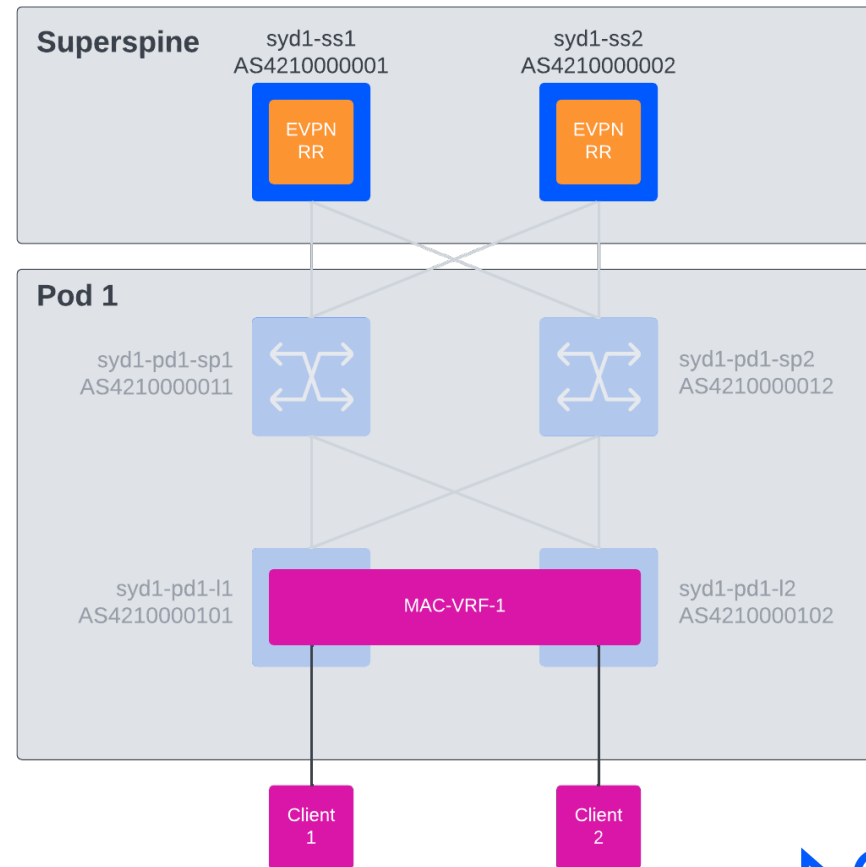
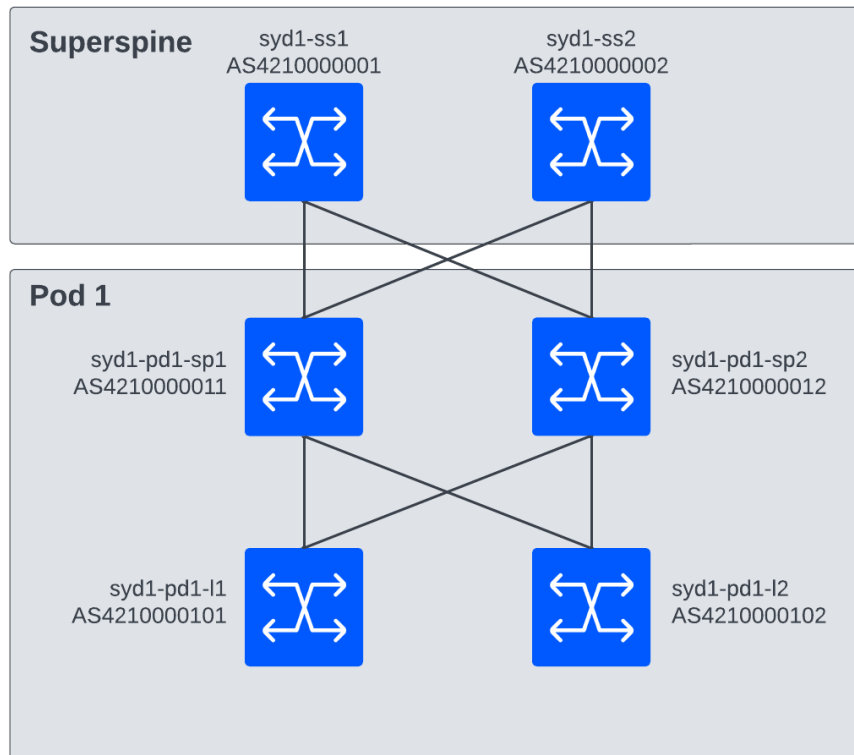


<https://github.com/network-unit-testing-system>

```
- test_class: TestNapalmBgpNeighbors
test_data:
- host: dcfabric-syd1-pd1-l1
  is_enabled: true
  is_up: true
  local_as: 4210000101
  local_id: 10.0.0.24
  remote_as: 4210000011
  remote_id: 10.0.0.24
- host: dcfabric-syd1-pd1-l1
  is_enabled: true
  is_up: true
  local_as: 4210000101
  local_id: 10.0.0.36
  remote_as: 4210000012
  remote_id: 10.0.0.36
```

# Example Network

## SRLinux-based Data Centre Fabric – EVPN-VXLAN



Let's do some Network Engineering and test along the way!

Live Demo 

# Takeaways

- Testing methodology isn't just about software development.
- Iteration of ideas, designs and processes need feedback-loops, testing concepts can enable this.
- Improve operational process, especially for repetitive tasks.
- Catch problems before they occur, especially if they've occurred before!
- Improve your design process and create a framework for how you think about network design.



Questions?

NOKIA

# Static Demo Slides

# Containerlab Network

```
(venv) root@containerlab:/home/tim/ausnog-2024# clab inspect
INFO[0000] Parsing & checking topology file: dcfabric.clab.yaml
```

#	Name	Container ID	Image	Kind	State	IPv4 Address	IPv6 Address
1	dcfabric-graphite	1e8cc9abdaa7	netreplica/graphite:latest	linux	running	172.20.20.12/24	2001:172:20:20::c/64
2	dcfabric-syd1-client-l1	93ac8297d594	ghcr.io/hellt/network-multitool	linux	running	172.20.20.19/24	2001:172:20:20::13/64
3	dcfabric-syd1-client-l2	e2fdf5f68a60	ghcr.io/hellt/network-multitool	linux	running	172.20.20.11/24	2001:172:20:20::b/64
4	dcfabric-syd1-pd1-l1	e70807910d34	ghcr.io/nokia/srlinux:latest	srl	running	172.20.20.20/24	2001:172:20:20::14/64
5	dcfabric-syd1-pd1-l2	074e64af64ea	ghcr.io/nokia/srlinux:latest	srl	running	172.20.20.17/24	2001:172:20:20::11/64
6	dcfabric-syd1-pd1-sp1	8a0e8511bc98	ghcr.io/nokia/srlinux:latest	srl	running	172.20.20.16/24	2001:172:20:20::10/64
7	dcfabric-syd1-pd1-sp2	a1a1266dddf6	ghcr.io/nokia/srlinux:latest	srl	running	172.20.20.18/24	2001:172:20:20::12/64
8	dcfabric-syd1-ss1	ad7bd1de0614	ghcr.io/nokia/srlinux:latest	srl	running	172.20.20.21/24	2001:172:20:20::15/64
9	dcfabric-syd1-ss2	cfdb1fd28f71	ghcr.io/nokia/srlinux:latest	srl	running	172.20.20.22/24	2001:172:20:20::16/64



## Example Tests

```
- test_class: TestNapalmLdpNeighbors
  test_data:
    - host: dcfabric-syd1-pd1-l1
      local_port: ethernet-1/27
      remote_host: syd1-pd1-sp2
      remote_port: ethernet-1/1
    - host: dcfabric-syd1-pd1-l1
      local_port: ethernet-1/28
      remote_host: syd1-pd1-sp1
      remote_port: ethernet-1/1
```

```
- test_class: TestNapalmBgpNeighbors
  test_data:
    - host: dcfabric-syd1-pd1-l1
      is_enabled: true
      is_up: true
      local_as: 4210000101
      local_id: 10.0.0.24
      remote_as: 4210000011
      remote_id: 10.0.0.24
    - host: dcfabric-syd1-pd1-l1
      is_enabled: true
      is_up: true
      local_as: 4210000101
      local_id: 10.0.0.36
      remote_as: 4210000012
      remote_id: 10.0.0.36
```

## Test Run – LLDP Neighbours

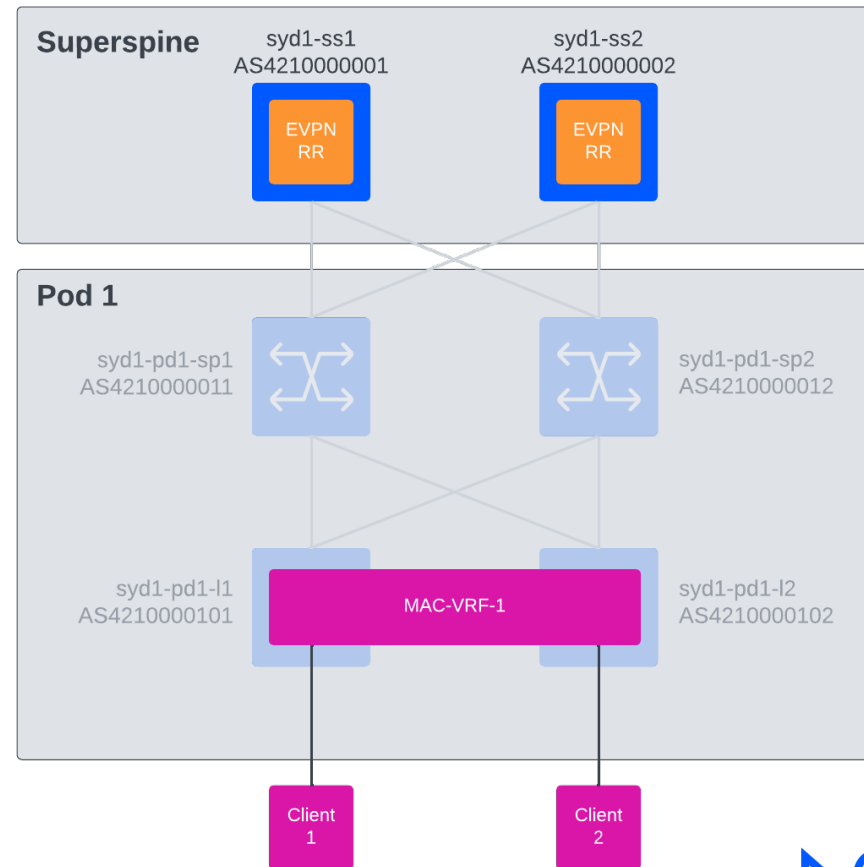
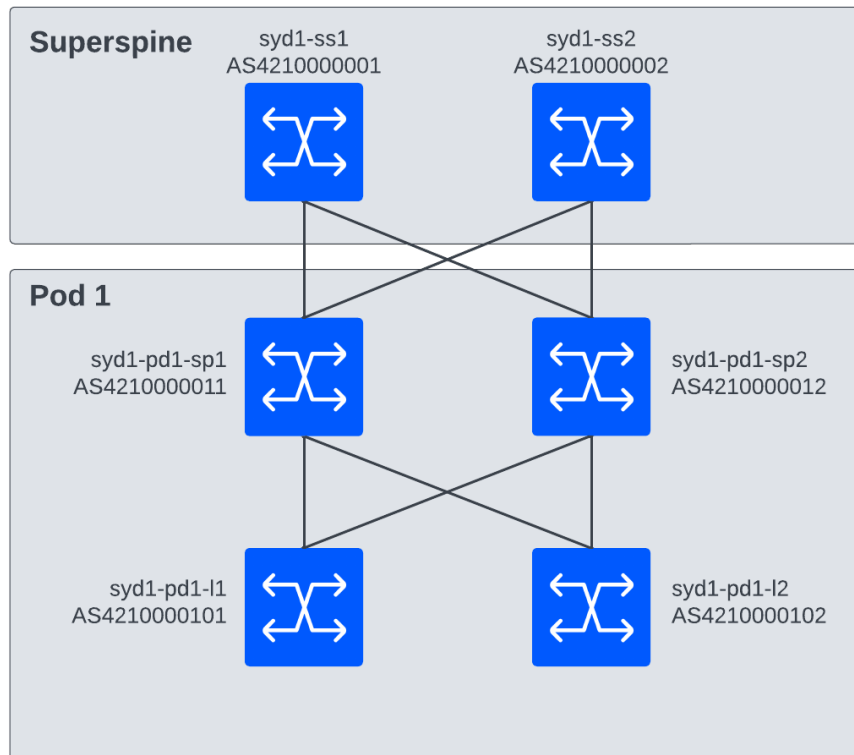
```
(venv) root@containerlab:/home/tim/ausnog-2024/nuts# pytest -v ./tests/test_basic_lldp_neighbours.yaml
Test session starts (platform: linux, Python 3.10.12, pytest 7.4.4, pytest-sugar 1.0.0)
cachedir: .pytest_cache
rootdir: /home/tim/ausnog-2024/nuts
configfile: pytest.ini
plugins: nuts-3.4.0, anyio-4.4.0, sugar-1.0.0
collected 4 items

../venv/lib/python3.10/site-packages/nuts/base_tests/napalm_lldp_neighbors.py::TestNapalmLldpNeighbors.test_remote_host[dcfabric-syd1-pd1-l1_0] ✓ 25%
../venv/lib/python3.10/site-packages/nuts/base_tests/napalm_lldp_neighbors.py::TestNapalmLldpNeighbors.test_remote_host[dcfabric-syd1-pd1-l1_1] ✓ 50%
../venv/lib/python3.10/site-packages/nuts/base_tests/napalm_lldp_neighbors.py::TestNapalmLldpNeighbors.test_remote_port[dcfabric-syd1-pd1-l1_0] ✓ 75%
../venv/lib/python3.10/site-packages/nuts/base_tests/napalm_lldp_neighbors.py::TestNapalmLldpNeighbors.test_remote_port[dcfabric-syd1-pd1-l1_1] ✓ 100%

Results (1.78s):
  4 passed
(venv) root@containerlab:/home/tim/ausnog-2024/nuts#
```

# Example Network

## SRLinux-based Data Centre Fabric – EVPN-VXLAN



## Pre-State - Building

```
# if "__name__" == "__main__":
hosts = ["dcfabric-syd1-pd1-l1", "dcfabric-syd1-pd1-l2", "dcfabric-syd1-pd1-sp1",
        "dcfabric-syd1-pd1-sp2", "dcfabric-syd1-ss1", "dcfabric-syd1-ss2"]
driver = get_network_driver('srl')

tests = []

for host in hosts:
    device = driver(host, 'admin', 'NokiaSrl1!', 10, {"jsonrpc_port": 80, "insecure": True})
    device.open()
    bgp_neighbors = device.get_bgp_neighbors()
    network_interfaces = ["ethernet-1/1", "ethernet-1/2", "ethernet-1/27", "ethernet-1/28"]
    interfaces = {k: v for k, v in device.get_interfaces().items() if k in network_interfaces}
    lldp_neighbors = device.get_lldp_neighbors()
    device.close()

    tests.append(build_nuts_bgp_neighbours_tests(bgp_neighbors, device))
    tests.append(build_nuts_interface_tests(interfaces, device))
    tests.append(build_nuts_lldp_neighbors_tests(lldp_neighbors, device))

print(yaml.dump(tests))
```

## Pre-State - Output

```
test_class: TestNapalmBgpNeighbors
test_data:
- host: dcfabric-syd1-pd1-l1
  is_enabled: true
  is_up: true
  local_as: 4210000101
  local_id: 172.20.1.101
  peer: 10.0.0.24
  remote_as: 4210000011
  remote_id: 10.0.0.24
- host: dcfabric-syd1-pd1-l1
  is_enabled: true
  is_up: true
  local_as: 4210000101
  local_id: 172.20.1.101
  peer: 10.0.0.36
  remote_as: 4210000012
  remote_id: 10.0.0.36
- host: dcfabric-syd1-pd1-l1
  is_enabled: true
  is_up: true
  local_as: 4200000000
  local_id: 172.20.1.101
  peer: 172.20.0.11
  remote_as: 4200000000
  remote_id: 172.20.0.11
- host: dcfabric-syd1-pd1-l1
  is_enabled: true
  is_up: true
  local_as: 4200000000
  local_id: 172.20.1.101
  peer: 172.20.0.12
  remote_as: 4200000000
  remote_id: 172.20.0.12
- test_class: TestNapalmInterfaces
test_data:
"/test_generated_prestate_fabric.yaml" 461L, 11191B
```

```
- test_class: TestNapalmInterfaces
test_data:
- host: dcfabric-syd1-pd1-l1
  is_enabled: true
  is_up: true
  mtu: 9232
  name: ethernet-1/1
  speed: 100000000.0
- host: dcfabric-syd1-pd1-l1
  is_enabled: false
  is_up: false
  mtu: null
  name: ethernet-1/2
  speed: 100000000.0
- host: dcfabric-syd1-pd1-l1
  is_enabled: true
  is_up: true
  mtu: 9232
  name: ethernet-1/27
  speed: 100000000.0
- host: dcfabric-syd1-pd1-l1
  is_enabled: true
  is_up: true
  mtu: 9232
  name: ethernet-1/28
  speed: 100000000.0
- test_class: TestNapalmLldpNeighbors
test_data:
- host: dcfabric-syd1-pd1-l1
  local_port: ethernet-1/27
  remote_host: syd1-pd1-sp2
  remote_port: ethernet-1/1
- host: dcfabric-syd1-pd1-l1
  local_port: ethernet-1/28
  remote_host: syd1-pd1-sp1
  remote_port: ethernet-1/1
```

## Pre-State – Test Run (1)

```
(venv) root@containerlab:/home/tim/ausnog-2024/nuts# pytest ./tests/test_generated_prestate_fabric.yaml
Test session starts (platform: linux, Python 3.10.12, pytest 7.4.4, pytest-sugar 1.0.0)
rootdir: /home/tim/ausnog-2024/nuts
configfile: pytest.ini
plugins: nuts-3.4.0, anyio-4.4.0, sugar-1.0.0
collected 308 items
```

## SRLinux Maintenance Groups

```
--{ + running }--[ ]--
A:syd1-pd1-sp1# info system maintenance
system {
  maintenance {
    group DRAIN_BGP_UNDERLAY {
      maintenance-profile DRAIN_BGP
      maintenance-mode {
        admin-state disable
      }
      members {
        bgp {
          network-instance default {
            peer-group [
              UNDERLAY
            ]
          }
        }
      }
    }
  }
  profile DRAIN_BGP {
    bgp {
      import-policy REJECT_ALL
      export-policy REJECT_ALL
    }
  }
}
--{ + running }--[ ]--
A:syd1-pd1-sp1# 
Current mode: + running
```

- NOS-integrated mechanism for applying configuration designed for maintenance
  - E.g. BGP Import/Export Policies
- Policies are completely configurable
- Multiple Maintenance Groups are possible
- One config item per group to enable or disable.

## Underlay BGP Neighbours

```
--{ + running }--[ ]--
A:syd1-pd1-sp1# show network-instance default protocols bgp neighbor

BGP neighbor summary for network-instance "default"
Flags: S static, D dynamic, L discovered by LLDP, B BFD enabled, - disabled, * slow
```

Net-Inst	Peer	Group	Flags	Peer-AS	State	Uptime	AFI/SAFI	[Rx/Active/Tx]
default	10.0.0.8	UNDERLAY	S	4210000001	established	28d:2h:48m:49s	ipv4-unicast	[1/1/3]
default	10.0.0.16	UNDERLAY	S	4210000002	established	28d:2h:22m:7s	ipv4-unicast	[1/1/3]
default	10.0.0.25	UNDERLAY	S	4210000101	established	10d:1h:12m:40s	ipv4-unicast	[1/1/3]
default	10.0.0.27	UNDERLAY	S	4210000102	established	28d:20h:1m:50s	ipv4-unicast	[1/1/3]

```
Summary:
4 configured neighbors, 4 configured sessions are established,0 disabled peers
0 dynamic peers
```

## Enabling Maintenance Group

```
--{ + running }--[ ]--
A:syd1-pd1-sp1# enter candidate
--{ + candidate shared default }--[ ]--
A:syd1-pd1-sp1# set system maintenance group DRAIN_BGP_UNDERLAY maintenance-mode admin-state enable
--{ +* candidate shared default }--[ ]--
A:syd1-pd1-sp1# diff
system {
    maintenance {
        group DRAIN_BGP_UNDERLAY {
            maintenance-mode {
                admin-state disable
                admin-state enable
            }
        }
    }
}
--{ +* candidate shared default }--[ ]--
A:syd1-pd1-sp1# commit now
All changes have been committed. Leaving candidate mode.
--{ + running }--[ ]--
A:syd1-pd1-sp1# show network-instance default protocols bgp neighbor
```

BGP neighbor summary for network-instance "default"  
Flags: S static, D dynamic, L discovered by LLDP, B BFD enabled, - disabled, \* slow

Net-Inst	Peer	Group	Flags	Peer-AS	State	Uptime	AFI/SAFI	[Rx/Active/Tx]
default	10.0.0.8	UNDERLAY	S	4210000001	established	28d:2h:49m:43s	ipv4-unicast	[4/0/0]
default	10.0.0.16	UNDERLAY	S	4210000002	established	28d:2h:23m:1s	ipv4-unicast	[4/0/0]
default	10.0.0.25	UNDERLAY	S	4210000101	established	10d:1h:13m:34s	ipv4-unicast	[4/0/0]
default	10.0.0.27	UNDERLAY	S	4210000102	established	28d:20h:2m:44s	ipv4-unicast	[4/0/0]

## Neighbour Import Statement State

```
--{ + running }--[ ]--  
A:syd1-pd1-sp1# info from state network-instance default protocols bgp neighbor * import-policy  
network-instance default {  
  protocols {  
    bgp {  
      neighbor 10.0.0.8 {  
        import-policy REJECT_ALL  
      }  
      neighbor 10.0.0.16 {  
        import-policy REJECT_ALL  
      }  
      neighbor 10.0.0.26 {  
        import-policy REJECT_ALL  
      }  
      neighbor 10.0.0.27 {  
        import-policy REJECT_ALL  
      }  
    }  
  }  
}  
--{ + running }--[ ]--
```

## Pre-State Tests (post MG)

[illegible]

## Custom Tests

```
import pytest
from utils.device_utils import *

devices = [
    "dcfabric-syd1-pd1-l1",
    "dcfabric-syd1-pd1-l2",
    "dcfabric-syd1-pd1-sp1",
    "dcfabric-syd1-pd1-sp2",
    "dcfabric-syd1-ss1",
    "dcfabric-syd1-ss2",
]

@pytest.mark.parametrize('device', devices)
def test_maintenance_group_not_active(device):
    maint_groups = get_maintenance_groups(device)
    active_groups = 0
    for group in maint_groups:
        if group.get("maintenance-mode").get("admin-state") == "enable":
            active_groups += 1

    assert active_groups == 0
```

## Custom Tests

```
(venv) root@containerlab:/home/tim/ausnog-2024/nuts# pytest ./tests/test_maintenance_groups.py
Test session starts (platform: linux, Python 3.10.12, pytest 7.4.4, pytest-sugar 1.0.0)
rootdir: /home/tim/ausnog-2024/nuts
configfile: pytest.ini
plugins: nuts-3.4.0, anyio-4.4.0, sugar-1.0.0
collected 6 items

tests/test_maintenance_groups.py ✓✓ 33% ██████████

_____ test_maintenance_group_not_active[dcfabric-syd1-pd1-sp1] _____

device = 'dcfabric-syd1-pd1-sp1'

@pytest.mark.parametrize('device', devices)
def test_maintenance_group_not_active(device):
    maint_groups = get_maintenance_groups(device)
    active_groups = 0
    for group in maint_groups:
        if group.get("maintenance-mode").get("admin-state") == "enable":
            active_groups += 1

> assert active_groups == 0
E assert 1 == 0

tests/test_maintenance_groups.py:21: AssertionError

tests/test_maintenance_groups.py x✓✓✓ 100% ██████████
===== short test summary info =====
FAILED tests/test_maintenance_groups.py: test_maintenance_group_not_active[dcfabric-syd1-pd1-sp1] - assert 1 == 0

Results (1.30s):
  5 passed
  1 failed
 - tests/test_maintenance_groups.py:13 test_maintenance_group_not_active[dcfabric-syd1-pd1-sp1]
```

## What about the EVPN Service?

```
import pytest
import itertools
from utils.device_utils import *

devices = ["dcfabric-syd1-pd1-l1", "dcfabric-syd1-pd1-l2"]
evpn_mac_types = ["evpn", "learnt"]

@pytest.mark.parametrize('device,mac_type', itertools.product(devices, evpn_mac_types))
def test_evpn_bridge_table_learnt_macs(device, mac_type):
    bridge_table = get_bridge_table(device)
    mac_count = 0
    for mac in bridge_table:
        if mac.get("type") == mac_type:
            mac_count += 1

    assert mac_count > 0
```

```
(venv) root@containerlab:/home/tim/ausnog-2024/nuts# pytest ./tests/test_evpn_bridge_tables.py
Test session starts (platform: linux, Python 3.10.12, pytest 7.4.4, pytest-sugar 1.0.0)
rootdir: /home/tim/ausnog-2024/nuts
configfile: pytest.ini
plugins: nuts-3.4.0, anyio-4.4.0, sugar-1.0.0
collected 4 items

tests/test_evpn_bridge_tables.py ✓✓✓✓ 100% ██████████

Results (0.90s):
  4 passed
```

## Do Maintenance, Undrain

- Shut Interfaces on the box (fully isolate)
- Do your maintenance:
  - Software Upgrade
  - Hardware Swap (complete/partial)
  - Configuration Testing
  - Load Testing
- Load device back up, with maintenance group still active.
- Enable all interfaces\*
- Test for Maintenance Groups
- Test for EVPN Service MACs

```
(venv) root@containerlab:/home/tim/ausnog-2024/nuts# pytest ./tests/*.py
Test session starts (platform: linux, Python 3.10.12, pytest 7.4.4, pytest-sugar 1.0.0)
rootdir: /home/tim/ausnog-2024/nuts
configfile: pytest.ini
plugins: nuts-3.4.0, anyio-4.4.0, sugar-1.0.0
collected 10 items

tests/test_evpn_bridge_tables.py ✓✓✓✓
tests/test_maintenance_groups.py ✓✓✓✓✓

Results (2.00s):
  10 passed
```



All good right...?

## Run our pre-state tests

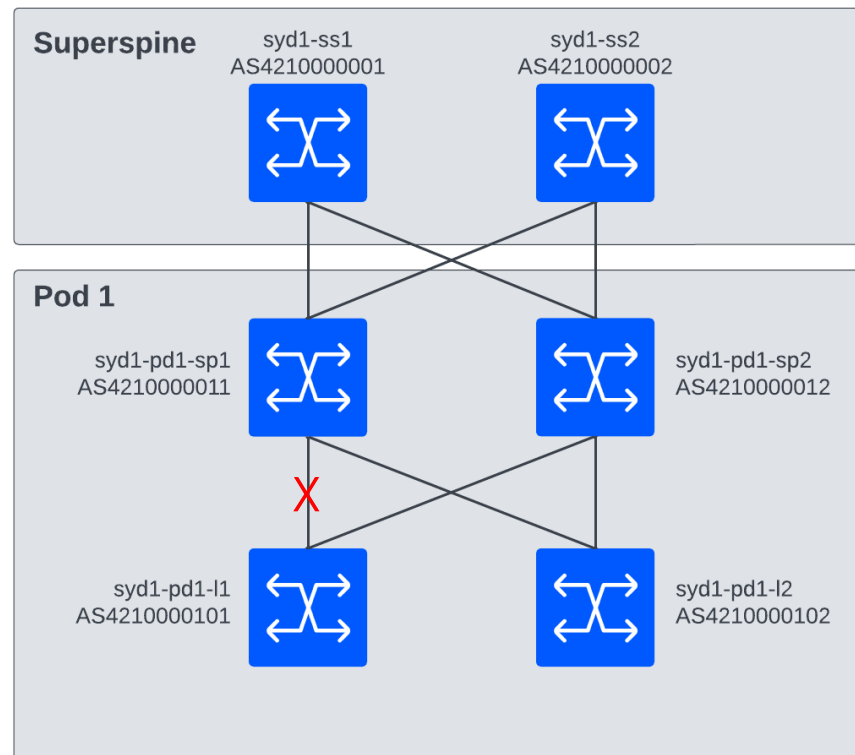
```
tests/test_generated_prestate_fabric.yaml x//////////SSSS////////// 68%
                                           //////////SSSS////////// 87%
                                           //////////SSSS////////// 100%
===== short test summary info =====
FAILED tests/test_generated_prestate_fabric.yaml::TestNapalmBgpNeighbors::test_is_up[dcfabric-syd1-pd1-l1_0] - assert False == True
FAILED tests/test_generated_prestate_fabric.yaml::TestNapalmInterfaces::test_is_up[dcfabric-syd1-pd1-l1_3] - assert False == True
FAILED tests/test_generated_prestate_fabric.yaml::TestNapalmLldpNeighbors::test_remote_host[dcfabric-syd1-pd1-l1_1] - KeyError: 'ethernet-1/28'
FAILED tests/test_generated_prestate_fabric.yaml::TestNapalmLldpNeighbors::test_remote_port[dcfabric-syd1-pd1-l1_1] - KeyError: 'ethernet-1/28'
FAILED tests/test_generated_prestate_fabric.yaml::TestNapalmBgpNeighbors::test_is_up[dcfabric-syd1-pd1-sp1_2] - assert False == True
FAILED tests/test_generated_prestate_fabric.yaml::TestNapalmInterfaces::test_is_enabled[dcfabric-syd1-pd1-sp1_0] - assert False == True
FAILED tests/test_generated_prestate_fabric.yaml::TestNapalmInterfaces::test_is_up[dcfabric-syd1-pd1-sp1_0] - assert False == True
FAILED tests/test_generated_prestate_fabric.yaml::TestNapalmLldpNeighbors::test_remote_host[dcfabric-syd1-pd1-sp1_0] - KeyError: 'ethernet-1/1'
FAILED tests/test_generated_prestate_fabric.yaml::TestNapalmLldpNeighbors::test_remote_port[dcfabric-syd1-pd1-sp1_0] - KeyError: 'ethernet-1/1'

Results (12.23s):
  275 passed
   9 failed
  24 skipped
- ../venv/lib/python3.10/site-packages/nuts/base_tests/napalm_bgp_neighbors.py:93 TestNapalmBgpNeighbors.test_is_up[dcfabric-syd1-pd1-l1_0]
- ../venv/lib/python3.10/site-packages/nuts/base_tests/napalm_interfaces.py:39 TestNapalmInterfaces.test_is_up[dcfabric-syd1-pd1-l1_3]
- ../venv/lib/python3.10/site-packages/nuts/base_tests/napalm_lldp_neighbors.py:53 TestNapalmLldpNeighbors.test_remote_host[dcfabric-syd1-pd1-l1_1]
- ../venv/lib/python3.10/site-packages/nuts/base_tests/napalm_lldp_neighbors.py:60 TestNapalmLldpNeighbors.test_remote_port[dcfabric-syd1-pd1-l1_1]
- ../venv/lib/python3.10/site-packages/nuts/base_tests/napalm_bgp_neighbors.py:93 TestNapalmBgpNeighbors.test_is_up[dcfabric-syd1-pd1-sp1_2]
- ../venv/lib/python3.10/site-packages/nuts/base_tests/napalm_interfaces.py:33 TestNapalmInterfaces.test_is_enabled[dcfabric-syd1-pd1-sp1_0]
- ../venv/lib/python3.10/site-packages/nuts/base_tests/napalm_interfaces.py:39 TestNapalmInterfaces.test_is_up[dcfabric-syd1-pd1-sp1_0]
- ../venv/lib/python3.10/site-packages/nuts/base_tests/napalm_lldp_neighbors.py:53 TestNapalmLldpNeighbors.test_remote_host[dcfabric-syd1-pd1-sp1_0]
- ../venv/lib/python3.10/site-packages/nuts/base_tests/napalm_lldp_neighbors.py:60 TestNapalmLldpNeighbors.test_remote_port[dcfabric-syd1-pd1-sp1_0]
```

Looks like the link between leaf1 and spine1 is down...

# Example Network

## SRLinux-based Data Centre Fabric – EVPN-VXLAN



## Passing Tests?

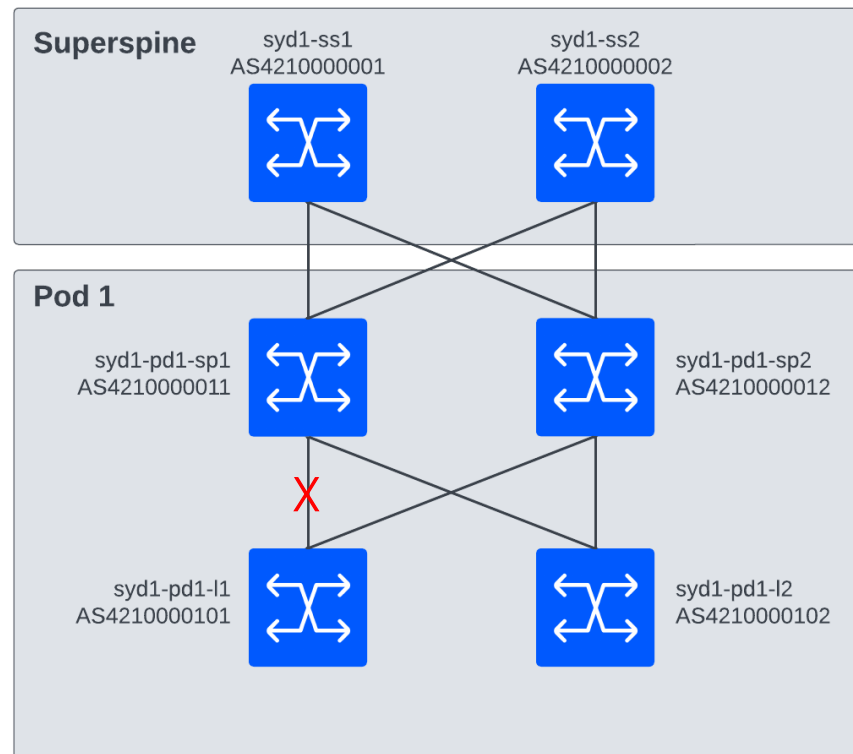
```
(venv) root@containerlab:/home/tim/ausnog-2024/nuts# pytest ./tests/*.py
Test session starts (platform: linux, Python 3.10.12, pytest 7.4.4, pytest-sugar 1.0.0)
rootdir: /home/tim/ausnog-2024/nuts
configfile: pytest.ini
plugins: nuts-3.4.0, anyio-4.4.0, sugar-1.0.0
collected 10 items

tests/test_evpn_bridge_tables.py ✓✓✓✓ 40% ██████████
tests/test_maintenance_groups.py ✓✓✓✓✓ 100% ██████████

Results (2.00s):
  10 passed
```

# Example Network

## SRLinux-based Data Centre Fabric – EVPN-VXLAN



## Fix the issue and re-run tests

```
--{ + candidate shared default }--[ ]--
A:syd1-pd1-sp1# set interface ethernet-1/1 admin-state enable
--{ +* candidate shared default }--[ ]--
A:syd1-pd1-sp1# diff
    interface ethernet-1/1 {
-        admin-state disable
+        admin-state enable
    }
--{ +* candidate shared default }--[ ]--
A:syd1-pd1-sp1# commit now
All changes have been committed. Leaving candidate mode.
--{ + running }--[ ]--
A:syd1-pd1-sp1# (venv) root@containerlab:/home/tim/
Current mode: + running Test session starts (platform: linux)
```

[illegible]

# Copyright and confidentiality

The contents of this document are proprietary and confidential property of Nokia. This document is provided subject to confidentiality obligations of the applicable agreement(s).

This document is intended for use by Nokia's customers and collaborators only for the purpose for which this document is submitted by Nokia. No part of this document may be reproduced or made available to the public or to any third party in any form or means without the prior written permission of Nokia. This document is to be used by properly trained professional personnel. Any use of the contents in this document is limited strictly to the use(s) specifically created in the applicable agreement(s) under which the document is submitted. The user of this document may voluntarily provide suggestions, comments or other feedback to Nokia in respect of the contents of this document ("Feedback"). Such Feedback may be used in Nokia products and

related specifications or other documentation. Accordingly, if the user of this document gives Nokia Feedback on the contents of this document, Nokia may freely use, disclose, reproduce, license, distribute and otherwise commercialize the feedback in any Nokia product, technology, service, specification or other documentation.

Nokia operates a policy of ongoing development. Nokia reserves the right to make changes and improvements to any of the products and/or services described in this document or withdraw this document at any time without prior notice.

The contents of this document are provided "as is". Except as required by applicable law, no warranties of any kind, either express or implied, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose, are made in relation to the accuracy, reliability or contents

of this document. NOKIA SHALL NOT BE RESPONSIBLE IN ANY EVENT FOR ERRORS IN THIS DOCUMENT or for any loss of data or income or any special, incidental, consequential, indirect or direct damages howsoever caused, that might arise from the use of this document or any contents of this document.

This document and the product(s) it describes are protected by copyright according to the applicable laws.

Nokia is a registered trademark of Nokia Corporation. Other product and company names mentioned herein may be trademarks or trade names of their respective owners.